Learning and Practice of the Property Specification Language

Nicolas Troquard
Laboratory for Applied Ontology (ISTC-CNR)

**Property Specification Language (PSL)** is the first industry-standard assertion language for modern hardware (and concurrent) system specification. The authors of *A Practical Introduction to PSL*, Cindy Eisner and Dana Fisman, played a prominent role in development of the language, which has been standardized (IEEE Std 1850-2005, *Property Specification Language*, revised in 2010).

Computer scientists have long viewed system behavior as an automaton, and in the hardware industry, a design is a symbolic description of this automaton. PSL is intended for design specification and documentation. Beyond the simple specification, PSL can serve during system simulation for dynamic verification. PSL is also mathematically rigorous. A purely formal language enables a design to be automatically, exhaustively verified. This static verification has been made possible with efficient model checking techniques developed over the past 30 years. Formal verification has had industrial success prior to PSL with, for example, Linear Temporal Logic (LTL), Sugar, and ForSpec; it wasn’t a coincidence that the ACM Turing Award went to the founding fathers of model checking in 2007.

PSL has four layers of specification: Boolean, temporal, modeling, and verification. Notably, the modeling layer is indeed a layer of specification. It doesn’t concern the design of the system but of the auxiliary signals that help express the system design’s temporal properties. The Boolean layer describes the basic system structure. The temporal layer describes a design’s temporal properties to be tested (e.g., every request is eventually acknowledged) and is built from elements of the Boolean layer and operators similar to temporal and dynamic logic. The verification layer contains the test instructions—the assertions of temporal properties the system must satisfy.

PSL comes in five flavors—to which the Boolean and modeling layers are subject—that correspond to the most popular design languages in industry: GDL, SystemC, SystemVerilog, Verilog-HDL, and VHDL.

The book’s focus is the temporal layer of PSL. The PSL temporal expressions are evaluated on system traces. The examples of traces are presented as waveforms that show rising and falling signals, which helps clarify the evaluation models and give them visual appeal. All of the book’s examples can be directly applied to the specification of actual industrial systems.

Chapter 2 (“Basic Temporal Properties”) and chapter 4 (“Weak vs. Strong Temporal Operators”) clearly explain the concepts underlying temporal operators, which readers will find invaluable. No formal semantics is provided, however.

Chapter 5 (“SERE Style”) explains the specification of temporal properties as (extended) regular expression. Many language additions and subtleties are possible. Although familiarity with regular expression is helpful, readers can nonetheless readily grasp the power of syntactic sugaring and the ease of writing intricate specifications.

Chapter 9 (“The Simple Subset”) identifies a temporal layer fragment containing classes of properties that are easier to understand and implement in a verification tool. In fact, it’s highly recommended that readers use this simple subset, because many tools...
don’t support PSL in its entirety. The simple subset is sufficiently expressive by itself—most engineers will typically need only this for most of their specifications.

Chapter 6 (“Clocks”) introduces special language constructs for clocks that capture the events of signals rising and falling. Modern semiconductor designs are typically highly modular, and clocks are necessary for synchronization. Chapter 14 (“Multiply-clocked Designs”) addresses both the practical aspects and the difficulties involved in specifying asynchronous systems. The language constructs for clocks are part of the Boolean layer, although that could be made clearer. The notion of a default clock requires the authors to discuss the verification layer, but within the context of the role clocks play in specifying temporal properties.

Beyond the fundamentals of the temporal layer, chapter 3 (“Some Philosophy”) presents a conceptual background. The chapter title is a bit misleading: rather than delving into philosophy per se, the chapter tries to clarify various notions such as properties, traces, clocks, and assertions. Chapter 12 (“More Philosophy—High-vs. Low-level Assertions”) presents three concrete simple systems and appropriate properties to assert about them, and is an excellent, highly useful chapter.

The authors suggest early in the book that some properties in the temporal layer have remarkable equivalences, yet a list of common equivalences is explicitly given only in chapter 11 (“Advanced Topics”). It’s an important chapter because it gives readers a more complete view on PSL. Section 11.4 is particularly welcome because it explains how the language operates over infinite traces, which are the proper support for static verification via model checking.

Chapter 13 (“Common Errors”) will help to dispel any confusion that some novice language users might have. The errors discussed here, however, are those that more experienced users who understand logic very well should not make (e.g., “oddsities” of the interaction between the operators of logical implication with other operands, or inappropriate weakness or strength of an assertion).

A Practical Introduction to PSL, unfortunately, is not a book for theoreticians—just four pages of bibliographic notes are dedicated to the history and the theory of the logics behind PSL. Especially noticeable by its absence is a discussion of the theoretical and pragmatic choices adopted by the IEEE standard proposal protagonists (see, e.g., http://www.ieda.org/ieee-1850). These choices were made to obtain a specification language that is as expressive as the modeling languages and that is intuitive to engineers. A discussion of PSL’s computational properties is also missing, which is a major drawback. The omission is particularly unfortunate because otherwise a system engineer might gain insight into what constitutes good specification practice. Good practice could make feasible a formal verification task that is apparently computationally intractable but, in fact, is not (much more information is available in “From Philosophical to Industrial Logics” by M.Y. Vardi; see, e.g., http://www.cs.rice.edu/~vardi/papers/icla09.pdf).

I must admit, I’d hoped for a treatment of PSL’s computational aspect more than an engineer typically would care in practice. An engineer will, however, miss a thorough presentation of the specification task explained within the context of system engineering. From the book’s contents, I doubt that readers will gain a clear understanding of the role that specification plays in the design cycle. A case study possessing more depth than the examples provided in chapter 12 would have been most helpful in this regard.

A Practical Introduction to PSL is essentially an informal presentation of the IEEE Std 1850-2005 Property Specification Language (PSL). The book presents many practical examples and is largely well-written. That said, however, the book is chiefly an engineering manual on PSL: it’s not a reference book for researchers or a textbook for students. The absence of tools for the general public somewhat makes this aspect a nonissue. But in my opinion it’s precisely the lack of well-presented theory, and the unavailability of tools, that show why, seven years after the IEEE standardization and five years after the book’s publication, the general academic world is focused elsewhere. PSL is not widely taught in computing science and electronic engineering schools. An exception, although admittedly in the research realm, is the 2009 paper by Dax et al. (“On Regular Temporal Logics with Past”; see, e.g., http://www.inf.ethz.ch/personal/felixkl/papers/ACTA.pdf).

Hardware verification is beginning to thrive in industry, yet well-trained students and some fresh ideas in PSL would be very welcome. Accordingly, we can hope that a second edition of A Practical Introduction would provide a more theoretical background for students, professors, and the research community.

Direct questions and comments about this review to Nicolas Troquard, Laboratory for Applied Ontology (ISTC-CNR), via alla Cascata 56/C, Povo, 38123 Trento, Italy; troquard@loa-cnr.it.